ASIC ECO 단계에서 효율적인 Timing Closure 방법론

서영호* · 최현준** · 유지상***· 김동욱**

An Efficient Timing Closure Methodology in ASIC ECO Step

Young-Ho Seo* · Hyun-Jun Choi** · Ji-Sang Yoo*** · Dong-Wook Kim**

이 논문은 2007년도 정부재원(교육인적자원부 학술연구조성사업비)으로 한국학술진흥재단의 지원을 받아 연구되었음(KRF-2008-331-D00335).

요 약

본 논문에서는 ASIC 기반으로 칩을 개발하는 경우에 ECO 단계에서 몇 가지 타이밍 위반을 효율적으로 수정할 수 있는 방법을 제안하고자 한다. 이러한 타이밍 위반은 여러 가지 원인으로 발생할 수가 있는데 이 원인들 중에서 툴들의 특성 때문에 발생하는 것이 주요인이다. 이러한 violation 중에서 가장 빈번히 발생하는 것이 셋업 시간 위반과 홀드 시간 위반이다. 먼저 이러한 타이밍 위반이 발생하는 원인을 분석한 후에 이들을 극복하기 위한 타이밍 조절 방법을 제안한다. 각각의 타이밍 위반들은 데이터 요구 시간을 증가시키거나 데이터 도달 시간을 감소시킴으로서 해결할 수 있는데 그 구체적인 방법들을 경우에 따라 제안한다. 이러한 방법들은 어떠한 정해진 알고리즘과 원리에 의해서 수행하기는 어렵고, 경우에 따라서 ASIC 엔지니어가 적절하게 선택하여 적용해야 한다.

ABSTRACT

In this paper, we propose an efficient methodology to fix timing violation in ECO step for ASIC process. Timing violation can occur from various reasons and the major cause is inconsistent correlation between EDA tools. The most frequent violation is setup time and hold time violation. First, we analyzed the reason of violation creation, and then proposed the adjusting method for overcome them. Each violation can be fixed by increasing data required time or decreasing data arrival time. We proposed the detailed technique on a case basis. It is difficult to execute these methods by routine of algorithm or principle. Therefore ASIC engineer needs to apply these technique to violation as conditions of the implemented design.

키워드

ASIC, ECO, Timing Closure, Setup time, Hold time, Violation

* 광운대학교 교양학부(yhseo@kw.ac.kr)
** 광운대학교 전자재료공학과(교신 : dwkim@kw.ac.kr)
*** 광운대학교 전자공학과

접수일자 2008. 10. 20

Ⅰ. 서 론

공정이 미세해질수록 ASIC 엔지니어가 물리적인 구현 시 고려해야 하는 사항들이 점점 많아지고 있다. 그러나 Time-to-market이 강조되면서 칩 개발기간은 점점 짧아지고 있다. 따라서 ASIC 엔지니어는 효율적이고 검증된 칩 개발 과정을 구축하고 각 과정마다 신뢰할 수 있는 EDA 툴(tool)을 사용해서 ASIC 칩을 개발하고 있다. 그러나 ASIC 칩을 제작하기 위한 모든 과정을 EDA 툴이 처리하여 줄 수는 없다. 툴마다 모델링 및 해석 방식에 차이가 있으므로 툴 들 간에 상관오차가 발생하게 되고, 이러한 오차들은 결국 타이밍 위반이라는 문제를 발생시킨다. 이러한 문제는 툴을 이용하여 대량의 타이밍 위반을 제거하여야 하겠지만 해결되지 않는 소량의 타이밍 위반은 ASIC 엔지니어가 수동적으로 ECO (Engineering Change Order) 작업을 수행해서 수정해야만 한다[1][2]. 타이밍 위반은 툴 간의 상관차이로 발생되는 타이밍 위반의 대표적인 예이다. 최적화 및 라우팅 과정에서 모든 툴이 시간적 제약을 만족시키는 것을 최우선시 하고 있지만 아직도 상관성 문제는 극복되지 않았고 한 회사에서 제공되는 툴을 사용하지 않는 한 극복될 수도 없을 것이다[3][4]. 따라서 ASIC 엔지니어는 STA(static timing analysis)[5][6]결과를 참고하여 수동으로 타이밍 위반을 수정해야 한다.

본 논문에서는 타이밍 위반 유형을 경우별로 제시하고 효율적으로 타이밍 위반을 수정하는 방법을 제안하였다. STA 툴은 Synopsys의 Prime-Time[5][6]을 사용하였으며 STA 조건은 별도로 언급하지 않는다.

Ⅱ. Manual Timing Closure

논리합성(Logic synthesis) 및 물리적인 합성(physical synthesis), PPO (post-placement optimization), 및 PRO (post-route optimization)[5]등의 노력에도 불구하고 언제나 STA 결과에는 셋업 시간 위반 및 홀드 시간 위반이 존재하게 마련이다. 물리적 구현[7] 과정에서 설계자의 실수가 있었을 수도 있고, 제약(constraint)이 과도하게 주어졌을 수도 있다. 타이밍 위반의 수가 많을 경우에는 먼저 툴을 이용하여 대다수의 타이밍 위반을 제거해야만 한다. 그러나 모든 타이밍 위반 slack이 툴에 의해 제거되지 않는다. STA 결과는 RC 추출, 지연 계산 등 전문적인 각각의 툴에서 산출된 결과를 종합하고 해석한 결과이다. 만약 이 모든 툴과 완벽하게 호환성이 있는 툴이라면 한 번에 타이밍 위반이 수정될 수 있다. 그러나 현실적으로 그러한 호환성을 보유한 툴은 존재하지 않는다. 결국은 설계자가 직접 STA 결과를 토대로 타이밍 위반을 수정해야만 한다. STA 자체도 매우 심도 있는 내용이므로 세밀한 내용은 참고문헌으로 대체하고 본 논문에서는 구체적으로 설명하지 않는다. 여기서는 실제 프로젝트 수행 시 필요한 부분만 요약해서 언급한다. 설명의 용이함을 위해서 특별한 언급이 없는 한 STA는 단일 코너의 일반 조건(typical condition)에서 수행되었다고 가정한다.

CTS 및 HFNS (high fanout net synthesis), 라우팅 [8][9][10] 등을 마치면 STA를 수행한다. 그 결과 타이밍 위반이 대량으로 발생하였으면 툴을 이용해서 이들을 수정하는 것이 효과적이다. 타이밍 위반이 발생한 모든 경로를 사람이 확인하며 수정하기에는 너무 많은 시간이 소요되기 때문이다. 표 1에서 처음 라우팅 과정을 마친 데이터베이스에 대한 STA 결과와 Astro[11] 및 Physical compiler를 이용해서 타이밍 위반을 수정한 후 STA 결과를 비교하였다. 표 1(WNS : worst negative slack)에서 툴을 이용함으로써 최대 5시간 내에 0.21ns 이상의 타이밍 위반을 수정하였음을 확인할 수 있다.

표 1. 첫 번째 타이밍 종결 결과
Table 1. first timing closure result

		Before	After	
			Astro	PC
Gate Count		1,970,293	2,310,522	2,095,561
Setup	WNS	-0.26	-0.09	-0.03
	#WNS	938	304	124
Run Time(H)		·	4.5	2.5

ASIC 엔지니어가 ECO 작업을 수행해서 표 1과 같은 결과를 얻기 위해서는 하루 이상의 시간이 소요된다. 이런 경우는 툴을 사용하는 것이 효과적이다. 그러나 툴로 모든 경로를 수정하지는 못한다. 앞에서 언급한 상관 차이로 인한 오차 때문이다. 표 1과 같이 300여개의 소량의 slack이 잔류한다면 앞으로 제안할 방법을 참고해서

ASIC 엔지니어가 직접 타이밍 위반을 수정하는 것이 효율적이다.

Ⅲ. 제안한 셋업 시간 위반 수정

3.1. 셋업 시간 분석

클록이 천이한 이후 다음에 천이하는 시간보다 데이터의 전파 경로에서 소요되는 지연시간이 커서 데이터가 먼저 안정화되어 있지 못하면 셋업 시간 위반이 발생한다. 셋업 시간 위반을 분석하기 위해서는 실제로 어떤 요소들이 영향을 미칠 수 있는지 고려해야 한다. 본 논문에서는 Synopsys의 Prime Time[6] 환경에서 경로 형태의 타이밍 해석 모델은 최대(max)로 설정하고 해석 및 실험하였다. 그림 1에는 실제 회로에서 발생하는 지연 요소들을 나타내었다.

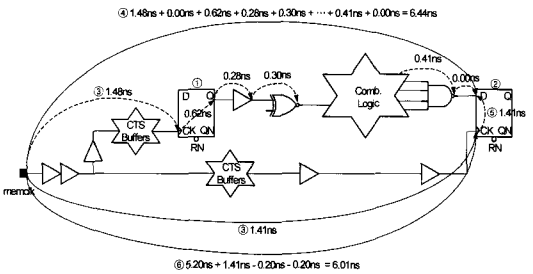

그림 1. 회로에서 발생하는 지연 요소.
Fig. 1. Delay factor in a example circuit.

앞에서 셋업 시간 위반이 발생하는 원인을 실제 회로 속에서 고찰해 보았다. 따라서 셋업 시간은 식 (1)과 같이 정의 될 수 있고 식 (2)의 조건에서 타이밍 위반이 발생하게 된다.

$$\textit{Setup time} = \textit{Data required time} - \textit{Data arrival time} \quad (1)$$

$$\textit{Data required time} < \textit{Data arrival time} \quad (2)$$

3.2. 데이터 요구 시간의 증가법

3.2.1. EP에서 클록의 전파 지연 증가

EP의 클록 핀에 인접해서 클록 버퍼를 삽입하는 것이 첫 번째 방법에 해당한다. 다른 CTS 경로 및 라우팅에 영향을 주지 않으면서 cell 지연만 타이밍 경로에 반영되도록 그림 3과 같이 EP의 클록 핀에 최대한 인접해서 삽입한다. 타이밍 위반이 미세한 경우 CTS 경로 상에 있는 클록 버퍼의 크기를 증가시킬 수도 있다. 이렇게 특정 CTS 경로의 skew를 조절하는 방법을 USC (useful skew control)이라 한다.

USC를 수행 전에는 항상 조절할 CTS 경로의 앞뒤 타이밍 경로를 살펴야 한다. 그림 3의 EP는 Q또는 QN으로 시작되는 또 다른 타이밍 경로의 SP가 되기 때문이다. 이들이 SP가 되면 그 경로의 데이터 요구 시간이 증가하므로 또 다른 셋업 시간 위반이 생길수도 있다. 또한 다음 타이밍 경로의 홀드 시간 위반을 유발할 수도 있다.

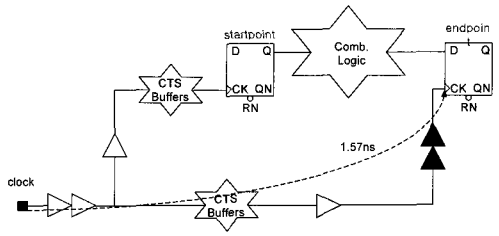

그림 2. 전파지연의 증가를 위한 USC의 적용.
Fig. 2. USC for increasing propagation delay.

3.2.2. 라이브러리 셋업 시간의 감소

일반적으로 빠른 순차 셀일수록 라이브러리 셋업 시간이 빠르다. 예를 들어서 QN 핀이 있는 순차 셀이 QN 핀이 없는 것보다 0.1ns 정도 빠른 라이브러리 셋업 시간을 가진다. 따라서 QN이 사용되는 경우라고 할지라도 라이브러리 셋업 시간을 감소시키고자 한다면 그림 3과 같이 QN 핀 쪽 경로에 인버터를 삽입하고 이를 Q 핀에 연결해야 한다.

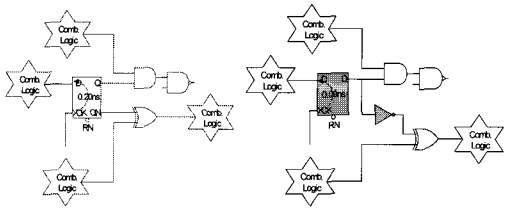

그림 3. 순차 셀의 변경을 위한 라이브러리 셋업 시간의 조절 (a) QN을 가진 셀 (b) QN이 없는 셀.
Fig. 3. Adjustment of library setup time by changing sequential cell (a) with QN port (b) without QN port.

앞의 경우와 같이 수정할 EP 셀은 다시 SP가 되므로 뒤쪽 타이밍 경로를 살펴본 후 신중하게 수정해야 한다. 또한 인버터를 삽입하고 이를 Q 핀으로 연결하지 않으면 동작 실패가 발생할 수 있으므로 formal check을 수행하여 올바르게 수정했는지 확인해야 한다.

3.3. 제안한 데이터 요구 시간의 감소법

3.3.1. SP에서 클록의 전파 지연 증가

앞에서 언급한 EP에서 "전파 지연 증가"에서와 비슷한 방법으로 SP의 클록 핀에 인접한 CTS 경로상의 클록 버퍼의 크기를 큰 것으로 교체하거나 제거한다. 이 방식을 그림 4에 나타내었다.

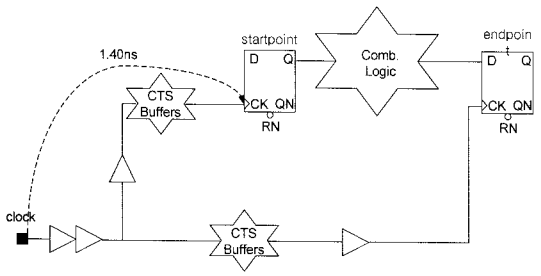

그림 4. SP에서 버퍼제거를 통한 전파지연 감소.
Fig 4. Decrease of propagation delay by removing buffer in SP.

3.3.2. 데이터 경로 지연 감소

SP의 Q 또는 QN 핀으로부터 EP의 D 핀까지는 무수히 많은 계층적인 레벨에 걸쳐 조합 셀들이 존재한다. 데이터 경로 지연은 이런 각각의 셀들의 지연시간들의 누적값이다. 따라서 각 셀들의 누적 지연이 계산되는 과정의 분석이 필요하다. 각각 셀들의 지연 시간은 두 가지 요소가 반영된 결과로써 이전 넷의 넷 지연과 셀의 입력 핀에서 출력 핀으로의 내부 지연이 반영된 결과이다. 그림 5에 넷 지연과 셀의 내부 지연이 결정되는 과정을 간단히 예시하였다.

앞의 내용을 근거로 데이터 도달 시간을 감소시키기 위한 방법은 아래와 같다.

1) 넷 천이와 캐패시턴스 감소법

일반적으로 셀의 증가 지연(incremental delay)은 입력 넷 천이 값과 출력 넷 캐패시턴스 값에 크게 의존한다. 입력 넷 천이는 이전과 현재 셀의 출력 핀의 천이와 입력 넷의 RC값에 크게 의존한다. 따라서 입력 넷 천이을 감소시키기 위한 방법은 아래와 같다.

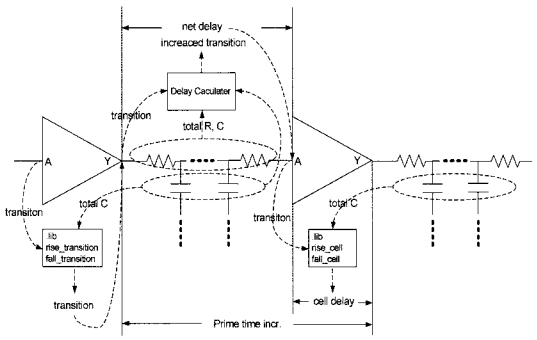

그림 5. 셀의 누적 지연 계산 방법.
Fig. 5. Calculation of incremental delay of cell.

① 이전과 현재 셀의 크기 증가

이 방법은 셀의 구동력(driving force)를 증가시켜 천이지연시간을 감소시키는 것이다. XOR, AND의 증가 지연이 크다고 가정하자. 문제가 되는 경로의 셀들에 대해 크기를 증가시킴으로써 버퍼의 구동력이 커져서 버퍼 출력신호의 천이시간이 감소한다. 따라서 XOR과 AND의 입력 천이지연시간이 감소한다. 그러므로 이들의 출력 천이지연시간도 감소하게 되고, 이들의 셀 증가 지연은 감소한다.

② 이전 셀의 출력에 버퍼삽입

셀의 크기 조정만으로 천이지연시간이 감소하지 않으면 신호의 세기를 증폭시키기 위하여 버퍼를 삽입한다. 문제가 되는 경로의 셀들은 더 이상 크기를 증가시킬 수 없을 정도로 크다고 가정한다. 버퍼와 XOR, AND간 거리가 먼 경우 이들 경로의 중간에 버퍼를 삽입한다. 버퍼(×20)로부터 발생한 신호의 에너지가 삽입된 버퍼(×20)에서 보상될 수 있다.

③ 이전과 현재 셀의 팬아웃 감소 및 분배

한 개의 셀이 너무 많은 팬아웃을 하면 천이지연시간이 증가한다. 이는 넷의 RC 값이 커지기 때문이다. 따라서 버퍼 등을 사용해서 팬아웃을 분배해야 한다. AND 셀이 64개의 셀을 구동하고 있고, 셀 증가 지연이 좋지 않다고 가정하자. 새로 만들어진 AND 셀의 앱력은 기존의 AND 셀과 같은 넷에서 연결한다. 이 경우에 버퍼를

삽입해서 팬아웃을 분배한다. 이 경로의 셋업 여유가 있다면 버퍼를 이용해서 분배하는 방법이 간단하다.

④ 현재 셀의 라우팅 경로의 조정

그림 6(a)는 라우팅 경로가 너무 긴 경우의 예를 나타낸 것이다. 물리 합성 시 잘못된 경로로 선언되었거나 주위에 밀집한 회로지역이 많은 경우에 이렇게 위치될 수 있다. 그림 6(b)는 이러한 경로를 단축시킨 예로써 라우팅이 길어지면 넷의 RC값이 커져서 넷 지연이 커지게 된다. 넷 지연에 의한 라우팅 효과도 무시할 정도는 아니다. 그림 6(c)는 아무 이유 없이 라우팅이 빙빙 도는 경우이다. Timing closure 과정이 반복되는 동안 ECO 라우팅을 하게 된다. ECO 라우팅을 많이 한 경로들에 대해서 이러한 현상이 나타난다. 이때는 이러한 경로에 대한 라우팅을 지우고 그림 6(d)와 같이 ECO 라우팅을 다시 수행해야 한다.

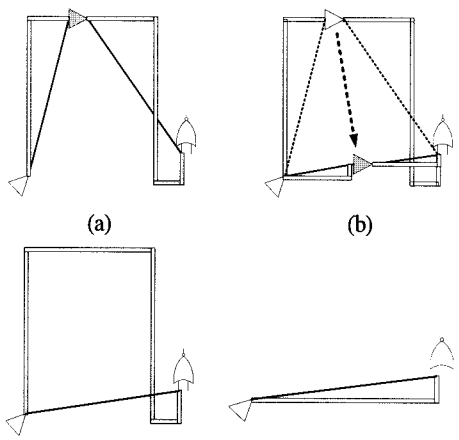

그림 6. Routing 경로의 수정 (a) 라우팅 예제1 (b) (a)의 수정 (c) 라우팅 예제2 (d) (c)의 수정.
Fig 6. Fixing of routing path (a) routing example (b) fixing of (a) (c) routing example2 (d) fixing of (c).

2) 계측정 논리 레벨 감소법

데이터 경로에 존재하는 몇 개의 셀 및 라우팅을 수정하기보다 논리 재맵핑 등의 방법을 이용해서 계층적인 논리 레벨을 줄이게 되면 타이밍 또는 면적, 전력 차원에서 큰 시간적인 여유를 확보할 수 있다. 그러나 오차가 존재할 경우에는 논리 실패의 결과를 초래하므로 수정 시 매우 신중해야 한다. 제안하고자 하는 논리 재맵핑 방법은 아래와 같다.

① 버퍼/인버터 사슬의 제거

시간적인 여유가 충분하거나 잘못된 경로로 선언되어 물리 합성된 부분에서 인버터/버퍼 사슬이 형성되는 경향이 있다. 최소 1개 이상의 버퍼 및 2개 이상의 인버터 등을 제거하게 되므로 타이밍 경로 레벨이 감소하게 되어 데이터 도달 시간이 감소한다. 가장 흔히 사용되는 논리 재맵핑 방법이다.

버퍼 사슬을 제거한 결과 각 셀들의 거리가 멀어져서 OR 셀의 입력 천이지연시간이 증가하게 되면 구동하는 셀의 크기를 증가시키던지 혹은 버퍼를 삽입해 주어야 한다. 주로 물리적인 셀의 입력/출력 포트쪽 경로에서 이런 경우가 많다.

② 고속 플립플롭으로의 변경

일반적인 경우에 플립플롭만 바꿔도 타이밍 위반을 수정 할 수 있는 경우가 대부분이다. 플립플롭을 교체하는 경우에 CTS시 조절된 skew가 틀어질 것이라고 우려할 수도 있다. 그러나 skew는 플립플롭의 클록 핀까지의 위상 지연만 고려하므로 셀 교체에 따른 큰 변동은 없다. 셀 마다 클록 핀의 캐패시턴스 차이는 존재하지만 이는 skew에 거의 영향이 없다. 앞에서 데이터 요구 시간의 라이브러리 셋업 시간을 감소시키기 위하여 EP의 플립플롭을 빠른 플립플롭으로 바꿀 때와 비슷한 방법이다. 한 타이밍 경로에서 SP와 EP의 플립플롭을 동시에 교체하면 최소한 0.3ns의 여유시간이 생긴다. 실험에 의하면 어떤 방법보다도 빠르고 효과적으로 셋업 시간 위반을 수정할 수 있다.

③ 버퍼/인버터 레벨의 조절

HFN(high fanout net) 합성 후에 인접해 있는 셀들 간 몇 단계의 버퍼/인버터를 거쳐 빙빙 돌아서 연결되는 경우가 있다. 대부분 타이밍에는 문제가 없으나 몇 개의 버퍼/인버터 레벨을 거치지 않고 직접 연결해준다면 제거된 레벨만큼 셀 지연이 감소하게 된다. HFN 외에도 데이터 경로 간 비효율적인 버퍼/인버터 레벨을 생략할 수 있는 경우가 많다.

④ 포트-포트 경로의 재합성

셀의 포트에서 포트간 논리 경로를 줄이는 것도 효율

적인 방법 중의 하나가 될 수 있다. 그림 7(a)와 같은 경로에서 셋업 시간 위반이 발생했다고 가정하자. 이는 매우 일반적인 예이다. 그림 7(b)에서 (d)까지 재맵핑을 통해서 경로를 감소하는 과정을 나타내었다.

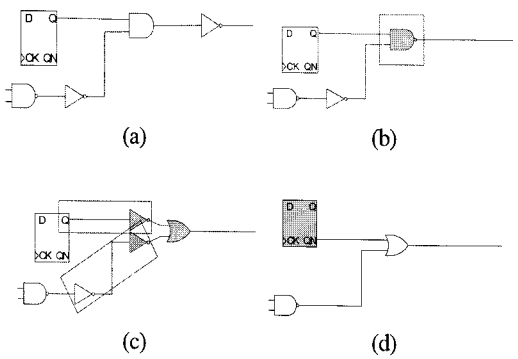

그림 7. 셋업 시간 위반 경로의 수정
(a) 원 회로 (b) NAND회로 (c) OR 회로
(d) QN 출력을 이용한 회로
Fig 7. re-mapping of setup time violated path
(a) original circuit (b) NAND circuit (c) OR circuit
(d) re-mapped circuit using QN port.

Ⅳ. 제안한 홀드 시간 위반 수정법

타이밍 위반을 수정하기 위해서는 셋업 시간이 어떻게 구성되는지 알아야 한다. 이를 근거로 타이밍 경로에서 왜 타이밍 위반이 발생했는지 원인을 명확하게 파악해야 한다.

4.1. 홀드 시간 분석

홀드 시간이란 클록이 천이되는 동안 입력된 데이터가 안정화 되어야 하는 최소한의 시간이다. 따라서 이전 데이터 경로에서 거의 지연시간이 소요되지 않아 클록의 천이시간 동안 입력된 데이터가 이전 데이터로 변하여 홀드 시간 위반이 발생한다. 그럼 일반적으로 홀드 시간 위반을 어떻게 검사하는지 따져볼 필요가 있다. 그림 8에서 이러한 타이밍 경로의 예를 나타내었다.

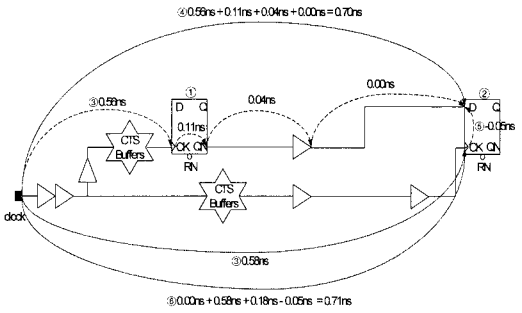

그림 8. 회로에서 발생하는 지연 요소.
Fig 8. Delay factor in a example circuit.

홀드 시간 위반은 식 (3)과 정의될 수 있고 식 (4)의 경우에 발생한다.

$$Hold\ time = Data\ arrival\ time - Data\ required\ time \quad (3)$$

$$Data\ required\ time < Data\ arrival\ time \quad (4)$$

4.1. 데이터 도달 시간의 증가법

4.2.1. SP에서 전파 지연의 증가

SP의 클록핀에 인접해서 클록 버퍼를 삽입한다. 다른 CTS 경로 및 라우팅의 영향없이 셀 지연만 타이밍 경로에 반영되도록 그림 9과 같이 클록핀에 최대한 인접해서 삽입한다. 타이밍 위반이 미세한 경우 CTS 경로상의 클록 버퍼를 증가시킬 수도 있다. 유용한 skew를 조절하기 전에는 항상 조절할 CTS 경로의 앞뒤 타이밍 경로를 살펴야 한다. 셋업 시간입장에서 데이터 도달 시간이 증가하게 되면 타이밍 위반을 유발시킬 수도 있기 때문이다. 일반적으로 홀드 시간 위반을 위해서 이렇게 까지 할 필요는 없다.

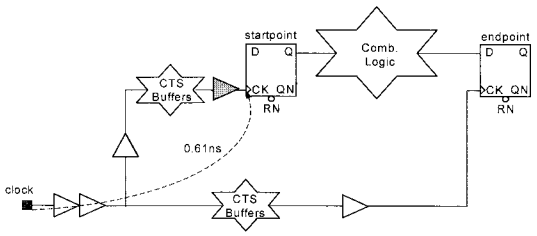

그림 9. SP에서 전파 지연의 증가를 위한 UCS
Fig 9. USC for increasing propagation delay in SP

1) 셀 삽입

가장 일반적으로 사용하는 방법이다. 그림 10과 같이 EP의 입력 바로 앞에 버퍼 또는 지연 셀을 삽입하여 데이터 도달 시간을 셀 지연만큼 증가시키는 방법이다.

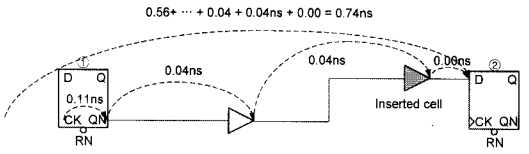

그림 10. 셀 삽입
Fig. 10. Cell insertion

2) 라우팅 경로의 조절

셀을 삽입할 공간이 없을 정도로 밀집도가 높은 회로에서는 그림 11와 같이 넷 지연을 증가시켜 라우팅 효과를 개선시키는 방법이 있다. 이 방법은 장점보다는 단점이 더 많을 수 있다. 라우팅이 빙빙 돌게 되면 라우팅 효율이 나빠진다. 또한 넷의 길이가 길어져서 넷 소모전력이 증가하게 된다. 따라서 극소수의 셀에 대해서만 사용해야할 방법이다.

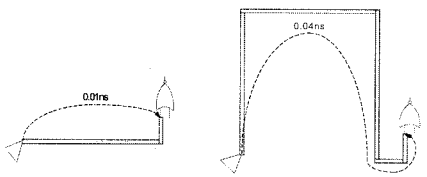

그림 11. 라우팅 경로의 수정 (a) 전 (b) 후
Fig. 11. Change of routing path (a) before (b) after

V. 실험 및 고찰

표 3에서 5까지는 앞에서 제안한 수동 timing closure 방법들에 대한 실험 결과 및 slack의 감소량을 나타내었다. 표 3을 살펴보면 라우팅 경로 조절에 의해서 천이지연시간이 0.8ns 감소한 것을 볼 수 있고 slack은 +0.03ns로 개선된 것을 볼 수 있다. 또한 팬아웃의 분리에 의해서도 1.5ns 천이지연시간이 감소한 것을 볼 수 있고 slack도 0.08ns로 개선된 것을 확인할 수 있다. 표 4에서는 버퍼 사슬을 제거하거나 플립플랍을 변경하는 방법을 이용하여 지연을 각각 0.22ns와 0.15ns 개선시킨 것을 볼 수 있고 slack도 개선시킬 수 있다는 것을 확인할 수 있었다. 그리고 표 5에는 표 1에 대한 두 번째 timing closure 결과를 나타내었다. 더 이상 툴이 slack을 수정하지 못함을 확인할 수 있었고, 표 1의 첫 번째 timing closure 후에 수동으로 직접 timing closure를 수행하면 고유 시간은 많이 소요되지만 slack을 대폭 감소시킬 수 있으며 게이트수를 유지 또는 소량 감소시킬 수 있음을 확인할 수 있다.

표 3. 천이지연시간에 대한 실허 결과.
Table 3. Experimental result for transition delay time.

	Routing Path Control		Fanout Separation	
	Before	After	Before	After
Delay time	1.20ns	0.40ns	2.60ns	1.10ns
Setup Slack	-0.05ns	+0.03ns	-0.07ns	+0.08ns

표 4. 예제 경로의 고유 지연에 대한 실험결과.
Table 4. Experimental result for intrinsic time of an example path.

	buffer Chain Removal		F/F Change	
	Before	After	Before	After
Delay	0.32ns	0.10ns	0.51ns	0.36ns
Setup Slack	-0.09ns	+0.13ns	-0.04ns	+0.11ns

표 5. 예제 경로의 고유 지연에 대한 실험결과.
Table 5. Second timing closure result of table 1

		After		
		Astro	PC	Manual
Gate count		2,330,099	2,150,112	2,097,003
Setup	WNS	-0.11	-0.04	-0.02
	#WNS	411	99	19
Run Time(H)		2.5	2	6.5

VI. 결론

본 논문에서는 ASIC 기반으로 칩을 개발하는 경우에

ECO 단계에서 몇 가지 타이밍 위반을 효율적으로 수정할 수 있는 방법을 제안하였다. 실험 결과를 보면 툴에 의해 먼저 timing closure 후에 수동으로 timing closure를 수행하면 시간은 많이 소요되지만 slack을 대폭 감소시킬 수 있으며 게이트수를 유지하거나 혹은 감소시킬 수 있음을 확인할 수 있었다. 따라서 제안한 방법을 이용하여 ASIC 설계 과정의 효율성을 높일 수 있을 것으로 사료된다.

참고문헌

[1] Olivier Coudert, "Timing and Design Closure in Physical Design Flows", ISQED 2002, pp. 511-516, Mar., 2002.

[2] Bart Vermeulen and Sandeep Kumar Goel, "Design for Debug: Catching Design Errors in Digital Chips", MDT2002, Vol. 19, pp. 35-43, May-Jun., 2002.

[3] R. Arunachalam, K. Rajagopal, L. Pileggi, "TACO: Timing Analysis with Coupling", Proc. of ICCAD''2000, Nov. 2000.

[4] G. Stenz, B.M. Reiss, B.Rohfleisch, F.M. Johannes, "Timing Driven Placement in Interaction with Netlist Transformations", in Proc. of ISPD''97, pp. 36-41, 1997.

[5] Himanshu Bhatnagar, Advanced ASIC Chip Synthesis Using Synopsys® Design CompilerTM and PrimeTime®, Kluwer Academic Publishers, 1999.

[6] Synopsys, Prime Time User Guide Advanced Timing Analysis, 2005.

[7] Synopsys, Physical Compiler User Guide, 2005.

[8] Nigel Horspool, and Peter Gorman, The ASIC Handbook, Prentice Hall, 2001.

[9] G. D. Hachtel, F. Somenzi, "Logic Synthesis and Verification Algorithms", Kluwer Academic Pub., 1996.

[10] H. Eisenmann, F. M. Johannes, "Generic Global Placement and Floorplanning", in Proc. of 35th DAC, June 1998.

[11] Synopsys, Astro User Guide, 2005.

저자소개

최현준(Hyun-Jun Choi)

2003년 2월 : 광운대학교 전자재료공학과 졸업(공학사)
2005년 2월 : 광운대학교 일반대학원 졸업(공학석사)
2005년 3월~현재 : 광운대학교 일반대학원 박사과정
※관심분야 : 영상압축, 워터마킹, 암호학, FPGA/ASIC 설계, Design Methodology

서영호(Young-Ho Seo)

1999년 2월 : 광운대학교 전자재료공학과 졸업(공학사)
2001년 2월 : 광운대학교 일반대학원 졸업(공학석사)
2000년 3월~2001년 12월 : 인티스닷컴(주) 연구원
2004년 8월 : 광운대학교 일반대학원 졸업 (공학박사)
2003년 6월~2004년 6월 : 한국전기연구원 연구원
2004년 12월~2005년 8월 : 유한대학 연구교수
2005년 9월~2008년 2월 : 한성대학교 전임강사
2008년 3월~현재 : 광운대학교 교양학부 조교수
※관심분야 : 2D/3D 영상 및 비디오 처리, 디지털 홀로그램, SoC 설계, 워터마킹/암호화

유지상(Ji-Sang Yoo)

1985년 2월 : 서울대학교 전자공학과 졸업(공학사)
1987년 2월 : 서울대학교 대학원 졸업(공학석사)
1993년 5월 : Purdue 대학교 전기공학과 졸업 (공학박사)
1993년 9월~1994년 8월 : 현대전자산업(주) 산전연구소 선임연구원
1994년 9월~1997년 8월 한림대학교 전자공학과 조교수
1997년 9월~현재 : 광운대학교 전자공학과 정교수
※관심분야 : 웨이블릿 기반 영상처리, 영상압축, 영상인식, 비선형 신호처리

김동욱(Dong-Wook Kim)

1983년 2월 : 한양대학교
전자공학과 졸업(공학사)
1985년 2월 : 한양대학교 대학원
졸업(공학석사)
1991년 9월 : Georgia공과대학 전기공학과 졸업
(공학박사)
1992년 3월~현재 : 광운대학교 전자재료공학과 정교수.
광운대학교 신기술 연구소 연구원
2000년 3월~2001년 12월 : 인티스닷컴(주) 연구원
※관심분야 : 디지털 VLSI Testability, VLSI CAD, DSP
설계, Wireless Communication